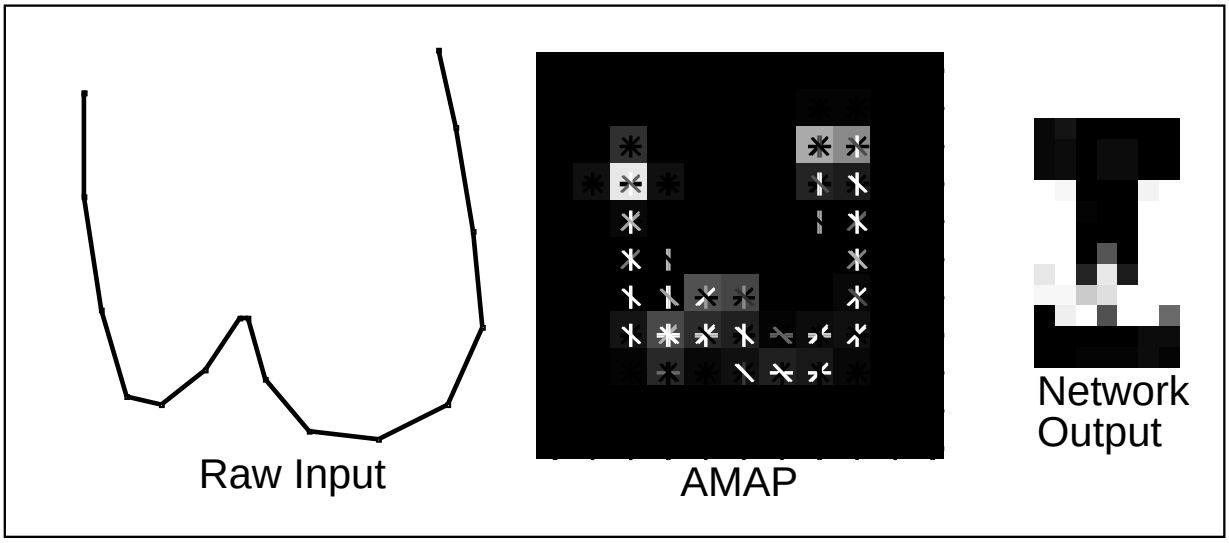

Figure 1: example of AMAP representation (center). The brightness of each pixel indicates the activity of the curvature detector. The brightness of each oriented bar indicates the activity in the corresponding angle plane. The Neural Net output code (right) uses bitmap representations of the characters.

CONCLUSION AND OUTLOOK

As pointed out earlier, the main advantage of the AMAP representation is that it makes very few assumptions about the nature of the input. It is designed to be applicable to any type of handwriting, including cursive. In the case of cursive handwriting or connected script, it is almost impossible to reliably segment a word into individual characters. Experiments are being performed with a system where the words are globally normalized. The AMAP for the whole word is computed and fed to a large "replicated" MLCNN which, instead of producing a single character output, produces a sequence of character candidates. The final answer is then extracted from the output using a dynamical word model (Matan et al., 1992).

References

Guyon, I., Albrecht, P., Le Cun, Y., Denker, J. S., and W., H. (1991). design of a neural network character recognizer for a touch terminal. *Pattern Recognition*, 24(2):105–119.

Le Cun, Y. (1986). Learning Processes in an Asymmetric Threshold Network. In Bienenstock, E., Fogelman-Soulié, F., and Weisbuch, G., editors, *Disordered systems and biological organization*, pages 233–240, Les Houches, France. Springer-Verlag.

Le Cun, Y. (1989). Generalization and Network Design Strategies. In Pfeifer, R., Schreter, Z., Fogelman, F., and Steels, L., editors, *Connectionism in Perspective*, Zurich, Switzerland. Elsevier. an extended version was published as a technical report of the University of Toronto.

Le Cun, Y., Matan, O., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D., and Baird, H. S. (1990). Handwritten Zip Code Recognition with Multilayer Networks. In IAPR, editor, *Proc. of the International Conference on Pattern Recognition*, Atlantic City. IEEE. invited paper.

Matan, O., Burges, C. J. C., LeCun, Y., and Denker, J. S. (1992). Multi-Digit Recognition Using a Space Displacement Neural Network. In Moody, J. M., Hanson, S. J., and Lippman, R. P., editors, *Neural Information Processing Systems*, volume 4. Morgan Kaufmann Publishers, San Mateo, CA.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In *Parallel distributed processing: Explorations in the microstructure of cognition*, volume I, pages 318–362. Bradford Books, Cambridge, MA.

Schenkel, M., Guyon, I., Weissman, H., and Nohl, C. (1993). TDNN Solutions for Recognizing On-Line Natural Handwriting. In *Advances in Neural Information Processing Systems 5*. Morgan Kaufman.

Tappert, C., Suen, C., and Wakahara, T. (1990). the state of the art in on-line handwriting recognition. *IEEE Trans. PAMI*, 12(8).

A FIRST SYSTEM

In this section, we describe the first of two series of experiments with on-line handwriting recognition. In this system, the AMAP was 4x8x8, with dimensions corresponding respectively to the orientation of the trajectory θ (quantized every 45 degrees), the horizontal and the vertical positions of the pen X, Y. The characters were size-normalized to a height of 6 pixels, and centered on the 8x8 AMAP.

The network had three layers of adaptive weights. The first hidden layer was of size 8x6x6 (eight feature maps of size 6x6), and each unit was connected to a 4x3x3 neighborhood in the input. The second hidden layer was 30x1x1, with each unit connected to the entire 8x6x6 layer below. The output layer was 26x1x1 (one for each class) with each unit connected to all 30 units in the previous layer. The full network had 20132 connections controlled by only 9772 free parameters because of the weight sharing in the first layer.

All the experiments were performed in a *writer independent* way: the sets of writers used for training and testing were disjoint. A first series of experiments was performed on a database of uppercase letters that had been written in boxes. There were 8900 training samples and 2000 test samples. The accuracy was 96.2% on the test set, and 99.2% on the training set. For the second series of experiments, the classifier was integrated into a word-reading system. With this system, users can write uppercase words without any constraint on the format. The character strings are first broken down into subcharacter units using heuristics. The units are then grouped into tentative characters in many different ways, and sent to the recognizer for scoring. A dynamic programming technique finds the best combined interpretations. These interpretations are optionally checked/corrected using a dictionary. Using feedback from the segmenter, the recognizer was trained, not only to recognize genuine characters, but also to reject non-characters. The first test database contained 222 four-letter English words. The performance was 79% word accuracy without dictionary, and 89% word accuracy with a 50000 word dictionary. Accuracies for 5-letter words were 70% and 89% respectively. On a second database of 606 three-letter to sixteen-letter words, the word accuracy was 83%. These performance figures are similar to the ones obtained with a carefully tuned Time Delay Neural Network that uses a temporal representation of the trajectory (Schenkel et al., 1993). A hybrid system that combines the AMAP recognizer and the TDNN yielded 89% word accuracy with a 50000 word dictionary. For this last database, writers were instructed to print a particular set of words using uppercase letters, nevertheless we estimate that about 5% of the words were either misspelled, written in lowercase or cursive, or unreadable. The performance improvement obtained by combining the two systems underscores the complementarity of their representations.

SECOND SYSTEM

An improved system was design to test the applicability of the AMAP representation to the full 95-class printable ASCII set. The AMAP was similar to the one described in the previous section, but it had an additional map whose elements contained estimates of the local curvature of the trajectory, and the presence of stroke ends. The full AMAP had 5 slices (1 curvature and 4 angles) of size 11x11 (to avoid boundary effects with the neural network). Figure 1 is a graphic representation of an AMAP.

The network had three layers of adaptive weights. The first hidden layer was 8x9x9, with connections to 5x3x3 neighborhoods on the input. The second hidden layer was 8x7x7 with connections to 8x3x3 neighborhoods on the previous layer. The output layer was 84x1x1 with 8x7x7 connections to the previous layer. Usually, each class is associated with the activation of a single output unit. However, this type of output code causes convergence problems when the number of classes is large. We decided to use a distributed code where each class is associated with a "bitmap" of the corresponding ASCII symbol on a 12x7 image (hence the 84 output units). Another rationale behind that choice is that characters that are confusable on the input will produce similar output configurations, thus allowing a postprocessor (lexicon or language model) to correct consistent errors.

The network was trained separately on uppercase alphabetic characters, lowercase characters, digits and ASCII symbols, all in writer independent mode. Performance on uppercase characters was 4.9% error on a test set of size 9122 after reaching 3.9% error on a training set of size 35402; on lowercase: 7.4% error with a test set of size 8201, 5.2% error on training set of size 32499; on digits: 1.25% error on a test of size 2938, 0.43% error on training set of size 11606; on symbols: 12.3% error on a test set of size 3960, 10.3% error on training set of size 15040 (many ASCII symbols are highly confusable). The total recognition time including the preprocessing is around 13ms on a Sun SparcStation 2.

ON-LINE HANDWRITING RECOGNITION WITH NEURAL NETWORKS: SPATIAL REPRESENTATION VERSUS TEMPORAL REPRESENTATION.

Yann Le Cun, Yoshua Bengio, Don Henderson, Anne Weisbuch, Haim Weissman, and Larry Jackel
AT&T Bell Laboratories, Holmdel, NJ 07733

AMAP

The recognition of handwritten characters from a pen trajectory on a digitizing surface is often done in the time domain. Trajectories are normalized, and local geometrical or dynamical features are sometimes extracted. The recognition is performed using curve matching (Tappert, Suen and Wakahara, 1990), or other classification techniques such as Neural Networks (Guyon et al., 1991). These approaches have many advantages, most notably the compactness of the representation. Unfortunately they tend to be sensitive to stroke ordering, and are not particularly well suited for use in integrated segmentation-recognition techniques. In addition, global geometric characteristics, such as whether a stroke crosses another stroke drawn at a different time, are not readily available in these types of representations.

Since the intent of the writer is to produce a legible *image*, it seems natural to preserve as much of the pictorial nature of the signal as possible, while at the same time exploit the dynamical information in the trajectory. We propose a representation scheme, called AMAP, where pen trajectories are represented by low-resolution images in which each picture element contains information about the local properties of the trajectory. More generally, an AMAP can be viewed as a function in a multidimensional space where each dimension is associated with a local property of the trajectory, say the direction of motion θ, the X position, and the Y position of the pen. The value of the function at a particular location (θ, X, Y) in the space represents a smooth version of the "density" of features in the trajectory that have values (θ, X, Y) (in the spirit of the generalized Hough transform). An AMAP is a multidimensional array (say 4x6x6) obtained by discretizing the feature density space into "boxes". Each array element is assigned a value equal to the integral of the feature density function over the corresponding box. In practice, an AMAP is computed as follows. At each sample on the trajectory, one computes the position of the pen (X, Y) and orientation of the motion θ (and possibly other features, such as the local curvature c). Each element in the AMAP is then incremented by the amount of the integral over the corresponding box of a predetermined *point-spread function* centered on the coordinates of the feature vector. The use of a smooth point-spread function (say a Gaussian) ensures that smooth deformations of the trajectory will correspond to smooth transformations of the AMAP.

A particularly useful feature of the AMAP representation is that it makes very few assumptions about the nature of the input trajectory. In particular, AMAPs can be used for all types of handwriting (capital, lower case, cursive, punctuation, symbols). Unlike many other representations (such as global features), AMAPs can be computed for complete words without requiring segmentation.

CONVOLUTIONAL NEURAL NETWORKS

Image-like representations such as AMAPs are particularly well suited for use in combination with MultiLayer Convolutional Neural Networks (MLCNN) (Le Cun, 1989; Le Cun et al., 1990). MLCNNs are feed-forward neural networks whose architectures are tailored for minimizing the sensitivity to translations, rotations, or distortions of the input image. They are trained with a variation of the Back-Propagation algorithm (Rumelhart, Hinton and Williams, 1986; Le Cun, 1986).

The units in MCLNNs are only connected to a local neighborhood in the previous layer. Each unit can be seen as a local feature detector whose function is determined by the learning procedure. Insensitivity to local transformations is built into the network architecture by constraining sets of units located at different places to use identical weight vectors, thereby forcing them to detect the same feature on different parts of the input. The outputs of the units at identical locations in different feature maps can be collectively thought of as a local feature vector. Features of increasing complexity and globality are extracted by the neurons in the successive layers. Configurations of the last layer are interpreted as character labels.

This weight-sharing technique has two interesting side effects. First, the number of free parameters in the system is greatly reduced since a large number of units share the same weights. Second, such a network can be applied to whole-word images without requiring prior segmentation (more on this later).